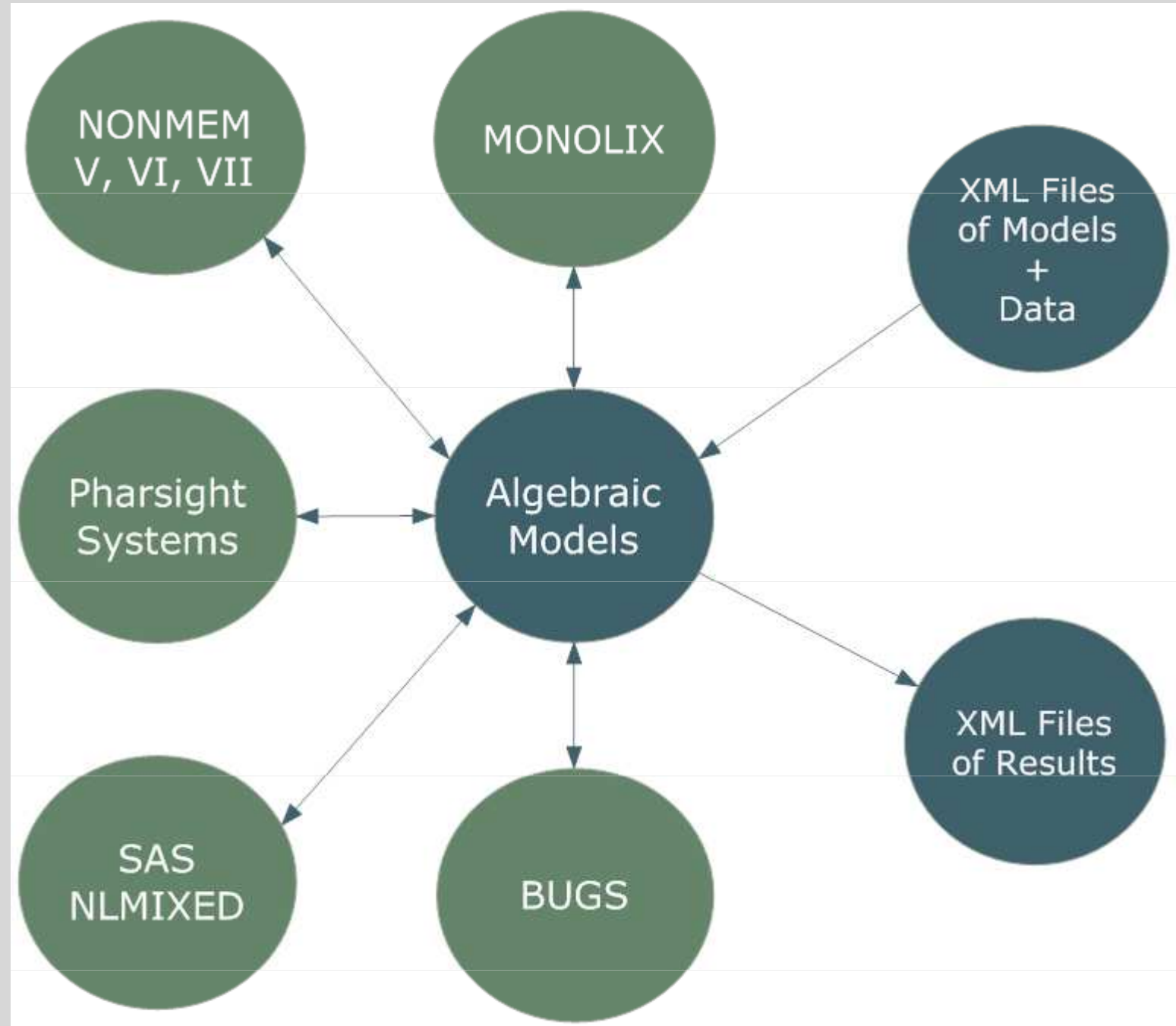


Open Statistical Services

Rich Haney, Cellular Statistics, LLC

Introduction

1. We work on “Open Statistical Services”. In this approach, systems of statistical models, as well as the processes by which we build and run them, are algebraic things.
2. In Pop PK/PD, we use a hub-and-spoke model for models as follows:



3. We hope to supplement public domain libraries of models with those culled from BUGS libraries, as well as models originally from Maple and other languages.
4. We would hope to complement “NLMEc-type” approaches.

What does this mean for individual models?

1. For internal use, we use StatML, with one file storing of equations for models + inputs + and results. This is in part to allow us to “compose” NONMEM, BUGS and other models together more easily.
2. We convert NONMEM, SAS PROC NLMIXED and other models and their data files to single files of equations. We do not use AND, OR, IF, THEN, ELSE, FOR LOOPS, or DO.
3. We say as much as we can about our models using polynomial-type forms. As such, we emphasize function composition, use of +, *, ^ (<= for us), D (for differentiation), and ` (for transpose or “adjoint” of a transformation from one polynomial basis to another.
4. We often go “back-and-forth” between algebraic representations and those involving use of true formal or informal languages.

Use of StatML for both Proteomics and Pop PK/PD

For both our proteomics and Pop PK/PD work, and for simplicity and modifiability in both cases, we use StatML. It is “flat”; in other words, it has no hierarchical structure. Some sample lines are:

```
<data model=1CPT_V.KA.CL state=Input" type="NONMEM">
  AMT(1): AMT <- [ 60000 60000 60000 60000 ..] </data>
<data model=1CPT_V.KA.CL state="Pred" type="NONMEM">
  DV(1):DV<- [ 38.5 100.7 ..246.8 1469.8 05.5 2.1] </data>
<equation model=...> ln ( DV<OBS> ) <- LNDV<OBS> </equation>
<equation model=...> InitialValues<A> <- [A1(0) = AMT , A2(0) = 0] </equation>
<equation model=...> DifferentialEqs<A> <- [ DA(1) = KA*A(1) ,
  DA(2)=KA*A(1)...] </equation>
```

As one might expect, we also use an equivalent representation that does not make use of XML wrappers, but instead is in the form of CSV files.

Notes

1. Instead of $IPred = .001$; If $F > 0$ then $IPred = \log(F)$, we use $IPred <- .001 + H(F) * (\log(F) - .001)$, as in BUGS
2. To representing multiple dosing, we use Dirac delta functions $DEQS<A> <- KA * A(1) + 6000 * (\delta(72) + \delta(106) + \delta(120))$.
3. We promote variables such as clearance (CL) to be types. These come with their own biology and equations, e.g.:

STRUCTURAL:PARAMETERIZATION <- $V \times KA \times CL$

Parameterizations - Algebraic Aspects

1. When showing parameterizations, we are substituting x (or $*$) where you might expect commas. We also typically get a second version of our equations at the same time. Instead of: $T <- [V, K, K12, K2]$ We use: $T <- V \times K \times K12 \times K21$ and $T \rightarrow V \times K \times K12 \times K2$
2. In the expression on the left, parameters are “going in” as inputs. On the right, we are saying that to be consistent, we take observations or make inferences to estimate the same parameters.
3. The first is an “assembly” of parts of some black box T ; the second involves using “lenses” to observe their behavior.
4. Some additional pairs of expressions add to our overall algebraic perspective on “static” versus “dynamic” systems.

The main question we are asking

1. We see if we can shed any light (or cannot shed any light) on systems for biological applications using the following pairs of approaches:

Item	Static Aspects	Behavioral/process Aspects
Biological Systems	Biological systems as algebras	Biological processes are coalgebras
Datasets	Datasets form dataset algebras	Systems successive runs and tools define coalgebras
Statistical models	Systems of models in a static sense	Taking observations or inferences using them is coalgebraic
Data types of programs	Abstract data types are algebras	Run-time aspects of programs define coalgebraic types (co-types)

2. At each level of abstraction, beginning with +, *, ^ (<=), `, and so on, we describe (and prescribe) what we need to do.
3. We make use our the theme of “going back and forth”, in this case back and forth between the above two types of pairs.

Algebraic operations at each level – a few examples

#	Operators for s A,B	Explanation using biological terms	Sample Meaning in system
1	<A,A>	Clone (Copy) – a crucial starting point (In algebra-speak, clone is one of the first “functors”)	Clone model A, e.g., to then make model B
2	A + B	View of A + B .. as members of a population, each of which is independent.	Create and view a set of models
3	A * B	View A and B as created by one parent (or are part of one overall “parent” structure)	View models as part of 1 parent concept
6	D(A)	A change in a continuous system, or structure such as a tree.	Notation used by us to track changes in model, using [6]
4	T T [~]	A back-and-forth flow, with T going one way, and T [~] going the other. (Here T [~] is the transpose or adjoint of T. T [~] is sometimes a pseudo-inverse.)	Translate NONMEM to BUGS, and then translate it back again to NONMEM.
6	A sim B	“Same behavior ” of two biological systems. (We model bioequivalence as probabilistic bismulation or co-simulation.)	Systems A and B have the same black-box behavior

We need about 18 more (as “functors”) for indexes (“fibers”), distributions, etc.

A + B..., where A, B... are models

Model Type(s)

NONMEM

Monolix

BUGS

Pharsight

CASL

Route

-- All --

Compartment

2CPT

Dosing

MD

TLAG

K

KA

Q

K12

K21

K31

Submit

Reset

Please select parameters to filter data:

Models	Inputs	Results	Study Name	Type	Routing	Cpt	Dosing	Parameters	TLAG	K	KA	Q
Models	Inputs	Results	BOLUS_2CPT_V.K.K12.K21.MD	Maple	Bolus	2CPT	MD	V.K.K12.K21				
Models	Inputs	Results	BOLUS_2CPT_V.K.K12.K21.MD	BUGS	Bolus	2CPT	MD	V.K.K12.K21				
Models	Inputs	Results	BOLUS_2CPT_V.K.V.M.M.K21.MD	Maple	Bolus	2CPT	MD	V.K.V.M.M.K21				
Models	Inputs	Results	BOLUS_2CPT_V.K.V.M.M.K21.MD	BUGS	Bolus	2CPT	MD	V.K.V.M.M.K21				
Models	Inputs	Results	BOLUS_2CPT_V1.V2.Q.CL.MD	Maple	Bolus	2CPT	MD	V1.V2.Q.CL				
Models	Inputs	Results	BOLUS_2CPT_V1.V2.Q.CL.MD	BUGS	Bolus	2CPT	MD	V1.V2.Q.CL				
Models	Inputs	Results	BOLUS_2CPT_V1.V2.Q.V.M.M.MD	Maple	Bolus	2CPT	MD	V1.V2.Q.V.M.M				
Models	Inputs	Results	BOLUS_2CPT_V1.V2.Q.V.M.M.MD	BUGS	Bolus	2CPT	MD	V1.V2.Q.V.M.M				
Models	Inputs	Results	INFUSION_2CPT_V.K.K12.K21.MD	Maple	Infusion	2CPT	MD	V.K.K12.K21				
Models	Inputs	Results	INFUSION_2CPT_V.K.K12.K21.MD	BUGS	Infusion	2CPT	MD	V.K.K12.K21				
Models	Inputs	Results	INFUSION_2CPT_V.K.V.M.M.K21.MD	Maple	Infusion	2CPT	MD	V.K.V.M.M.K21				
Models	Inputs	Results	INFUSION_2CPT_V.K.V.M.M.K21.MD	BUGS	Infusion	2CPT	MD	V.K.V.M.M.K21				

A * B..., when A, B,... are models

Much of the time we create and compare different types of BUGS models. In this case we compare NONMEM, Maple BUGS versions

4 #SORROUTINE ADVANI, TRANSI	4 k = phi(Iid.obs,2);	5 #Define dependencies
5 #PK	5	6 #declare(A1(c), extra(c), myDenom(t));
6 TVM=TBETA(1)	6 t_obs = X.obs(:,1);	7
7 k=TVV*EXP(ETA(1))	7 t_dose = X.dose(:,1);	8 #Define Assumptions
8 TVM=TBETA(2)	8 D = X.dose(:,2);	9 assume(
9 V=TVV*EXP(ETA(2))	9 B=K_B_dose;	10 0 < IVV
10 S1=V	10 k_dose = k(Iid.obs_dose);	11 ,0 < K
11 #ERROR	11	12 ,0 < TVX
12 IPRED=F	12 B(O,I_dose)=0.*exp	13 ,0 < V
13 V=V*EXP(ESR(1))	13 (k_dose.*t_dose);	14 ,0< t
14 #TBETA (0,0.075,0.5) (0,90,200)	13 CS=csunum(B,1);	15 ,t < 222
15 # K V	14 F=(exp(-k.*t_obs)/V).*CS	16 ,0 <= myDenom(t)
16 #OMEGA 0.25 0.15	15 (K.*t_dose);	17 ;
17 # IIV_K IIV_V	16	18 #Define initial conditions

Moving from “back-and-forth” to spiral processes

1. So, we start by going back-and-forth between notions of algebraic structure and practical languages. We use the low level steps to the left (beginning with cloning, +, and *); we also use those in back-and-forth ways as well. We do this for both static (algebraic) and behavioral (coalgebraic) descriptions.
2. From there we need to move to higher-level steps, done within overall “spiral” models of model development.[7]. (We have not yet started programming the higher-level steps yet, however.)

We see if we can shed light on Pop PK/PD

In general, we see if can shed light on Pop PK/PD using universal algebra (UA) and coalgebra. These originated in the 1930s.



The field was started in part by W. Birkhoff as mentored by Emmy Noether (to left)



We use work by F.W Lawvere, J. Lambek (to right), B. Jacobs and others from the 1970s on.

To us, category theory is now a crucial part of universal algebra.

Use of the word “Universal” ...

The word “Universal” (as in JANUS) or “Unified” (as in UML) are used often when talking about large systems for biological applications. We think both of those are great. However, we feel there is also a role for an approach where we “go direct”. In that approach, the “U” and “Universal” refer to the Universal in Universal algebra and coalgebra. An advantage of this approach is that we can use the same sets of perspectives, and also the same algebraic notation, from top to bottom.

Summary

1. Open Statistical Services in practice involves our helping curate new sets of models in Maple, BUGS, Monolix, and other languages. We also help companies test and deploy new PK tools
2. We consider systems of statistical models as algebraic things. We make use of universal algebra, which involves polynomial type operations (+, *, etc.), and a perspective on processes (as coalgebras). We believe this defines a useful, supplementary approach to discussing statistical systems over the long run.

References

- [1] Rich Haney, Informal presentation at SAS, April 2004
- [2] “Interoperability of MATLAB, SAS, and R: Formal Methods and Practical Aspects”, SUN Computational Biology Conference, September 22, 2005
- [3] Peter McCullagh, “What is a statistical model?”, The Annals of Statistics, 2002, Vol. 30, No. 5, 1225–1310
- [4] Seth Sullivant, “Statistical Models are Algebraic Varieties”, 2006, <http://www.math.harvard.edu/~seths/lecture1.pdf>
- [5] Rich Haney, “Category theory for use in formal systems design for biological applications: useful approach or Siren Song?”, 1985.
- [6] J. M. Rutten, “Behavioral differential equations: a coinductive calculus of streams, automata, and power series”, Theoretical Computer Science 308:3:1-53
- [7] Rich Haney “RAMP: Reasoning Using Algebraic Models of Processes”, 2006 (as job talk), and 2007, unpublished MS; these were versions of earlier “Use of a hexagonal model of knowledge”, 2002

Acknowledgements

Specifics of the idea of Open Statistical Services came in response to a meeting with the NONMEM consortium in September 2004 in which the author was involved.

Patrick McConnell and the author both worked on the initial versions of StatML. We also want to acknowledge initial funding of this work by caBIG (e.g., to include work on StatML).

Thor Mirchandani and Brad Sparks of Presens Technologies did the database and the model comparer. This lets us compare our hundreds of models.